CF 655 div2

A

水题，吓唬人的。

```
#include<iostream>
#include <cstdio>
#include <algorithm>
using namespace std;
int main()
{
    int t;
    cin >> t;
    while(t--)
    {
        int n;
        cin >> n;
        for (int i = 1; i <= n;i++)
            cout << 1 << ' ';
        cout << endl;
    }
    return 0;
}
```

B

给出一个 n，要求两个正整数 a 和 b，使得 a+b=n 并且 LCM(a, b) 尽量小。多组数据。

思路：找出 n 的大于 1 的最小因子 k，a = n/k,b = n - n/k，不会证明，感觉是对的。

```
#include<iostream>
#include <cstdio>
#include <algorithm>
#include<cmath>
using namespace std;
int main()
{
    int t;
    cin >> t;
    while(t--)
    {
        int n;
        cin >> n;
        int flag = 1;
        for (int i = 2; i <= sqrt(n);i++)
```

```
        {
            if((n/i)*i == n)
            {
                cout << n - (n / i)<<' '<<n/i<< endl;
                flag = 0;
                break;
            }
        }
        if(flag)
            cout << n - 1 <<' '<<1<< endl;
    }
    return 0;
}
```

C

对于一个序列，定义一个操作，选定连续的区间，将里面的数字重排，保证没有数字在原本的位置。现在给你一个 n 的排列 a□问你最少操作多少次，可以让 a 变成严格单增数列。

题解：显然，情况1：如果a本来就单调增，则为0次。情况2：如果没有一个数字在对应位置上，则为1次。情况3：在其他情况下，总能找到一个方法，对全部的区间重排，使得每个数字都不在自己原本的位置上并且不是最终要求的位置，这样就能转化为情况2，所以答案为2。

```
#include<iostream>
#include <cstdio>
#include <algorithm>
#include<cmath>
using namespace std;
const int MAX = 2e5 + 20;
int pic[MAX];
int main()
{
    int t;
    cin >> t;
    while(t--)
    {
        int n;
        cin >> n;
        int flag = 0,pos = -1;
        for (int i = 1; i <= n;i++)
        {
            scanf("%d", &pic[i]);
            if(pic[i]!=i && flag == 0)
                flag = 1,pos = i;
        }
        if(flag == 0)
        {
            cout << 0 << endl;
```

```
        }
        else
        {
            int posend = 0;
            for (int i = n; i >= 1;i--)
                if(pic[i]!=i)
                {
                    posend = i;
                    break;
                }
            for (int i = pos; i <= posend;i++)
            {
                if(pic[i] == i)
                {
                    flag = 114514;
                    break;
                }
            }
            if(flag == 114514)
            {
                cout<<2<<endl;
            }
            else
                cout << 1 << endl;
        }
    }
    return 0;
}
```

D

给定一个长度为奇数的环形区间，每次可以选择一个数，把这个数字替换成和这个数字相邻的两个数字的和，并删去相邻的两个数字，直到只剩下一个数字。求这个数字的最大值

题解：一开始想区间dp发现数据太大搞不了，又想优先队列贪心，后面又发现不太对。实际上，如果我进行了一次操作，得到了一个新数字a,那么我的下一次操作就最好不要选择与a相邻的数，因为这样的话，实际上相当于在原来的环上删掉了2个数字（因为a是原本环上两个数字 的和），按照这个思路贪心即可。实现的话，直接拆环就行。

```
#include <cstdio>
#include <iostream>
using namespace std;
typedef long long ll;
const int maxn = 4e5+55;
ll a[maxn];
ll c[maxn];
ll d[maxn];
int main(){
    int n;
    ll sum=0;
```

```
    scanf("%d",&n);
    for(int i=0;i<n;++i){
        scanf("%lld",&a[i]);
        sum+=a[i];
    }
    for(int i=0;i<n;++i){
        if(i>=2)c[i]=c[i-2]+a[i];
        else c[i]=a[i];
    }
    for(int i=n-1;i>=0;--i){
        if(i<=n-3)d[i]=d[i+2]+a[i];
        else d[i]=a[i];
    }
    ll maxx=c[0];
    for(int i=0;i<n;++i){
        maxx=max(maxx,c[i]+d[i+1]);
    }
    printf("%lld\n",maxx);
    return 0;
}
```